**Data Visualization Using Matplotlib and Seaborn**

**VISUALIZATION**

* Visualization is the process of representing data or information in a graphical or pictorial form. It helps to understand large amounts of data easily by showing patterns, trends, and relationships through visuals like charts, graphs, and maps.
* Instead of looking at raw numbers, visualization allows us to see data in a clear and meaningful way. It helps in making decisions faster and communicating information effectively. Common tools used for visualization include Microsoft Excel, Tableau, Power BI, and programming libraries like Matplotlib and Seaborn in Python.
* In simple terms, visualization turns data into pictures that make it easier to analyze, compare, and share insights.

**MATPLOTLIB**

* Matplotlib is a Python library used for creating graphs and charts from data. It helps to visualize data in an easy and understandable way. With Matplotlib, we can draw different types of charts like line charts, bar charts, pie charts, histograms, and scatter plots.
* It is mostly used in data analysis and machine learning projects to show patterns and trends in data. Matplotlib works well with other Python libraries like NumPy and Pandas.
* A plot of matplotlib contains:

\* Figure

\* Axes

\* Axis

\* Artists

**Figure**

* It is the overallcontainer or canvas that holds all plots and elements.
* A Figure can contain one or more Axes (plots) inside it.

**Axes**

* The area where data is actuallyplotted (contains the chart).
* Each Axes can have X and YAxis and multiple Artists (lines, labels, etc.).

**Axis**

* Represents the scale and direction of the data (X-axis and Y-axis).
* Controls limits, ticks, and labels for the plot.

**Artists**

* All visual elements like lines, text, titles, labels, and legends.
* Everything that appears on the plot is considered an Artist in Matplotlib.

**PYPLOT**

Pyplot is a module of Matplotlib that provides all the basic plotting functions in one place. It makes creating charts and graphs simple, such as line plots, bar charts, scatter plots, pie charts, histograms, and area charts.

To use Pyplot, we import it from Matplotlib like this:

* import matplotlib.pyplot as plt

We also import **NumPy** to help with numerical calculations and data handling:

* import numpy as np

**Some of the plots in Matplotlib:**

**LINE PLOT**

**Explanation:** Shows the relationship between two variables or trends over time.  
**Input:** Lists or arrays for X and Y values.

**Code Example:**

import matplotlib.pyplot as plt

x = [1, 2, 3, 4, 5]

y = [2, 4, 6, 8, 10]

plt.plot(x, y)

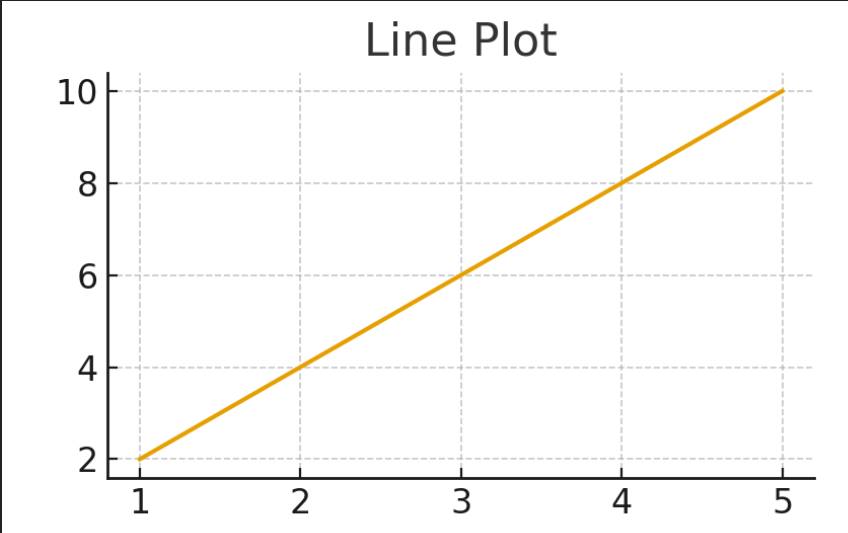
plt.title("Line Plot Example")

plt.xlabel("X-Axis")

plt.ylabel("Y-Axis")

plt.show()

**Output:** A simple line connecting points (1,2), (2,4), … showing an increasing trend.



**BAR CHART**

**Explanation:** Compares values across different categories.  
**Input:** Lists of categories and their corresponding values.

**Code Example:**

categories = ['A', 'B', 'C', 'D']

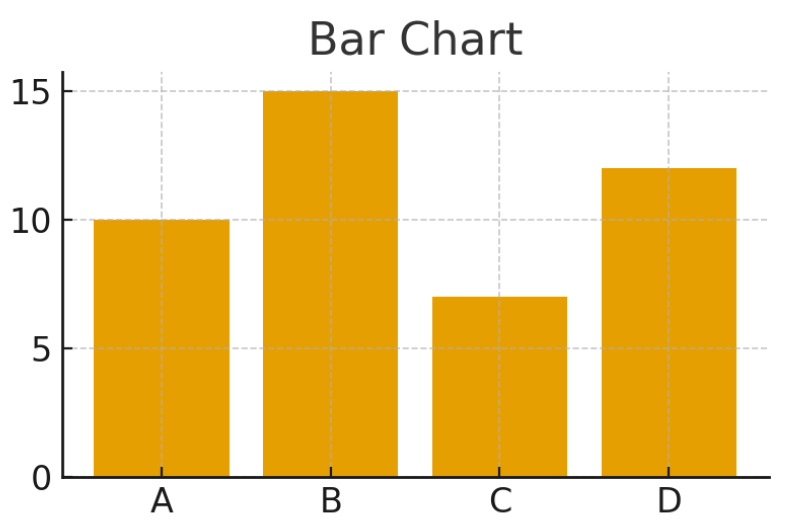
values = [10, 15, 7, 12]

plt.bar(categories, values)

plt.title("Bar Chart Example")

plt.show()

**Output:** Vertical bars representing the values of each category.



**HISTOGRAM**

**Explanation:** Shows frequency distribution of data in intervals (bins).  
**Input:** List or array of numerical data.

**Code Example:**

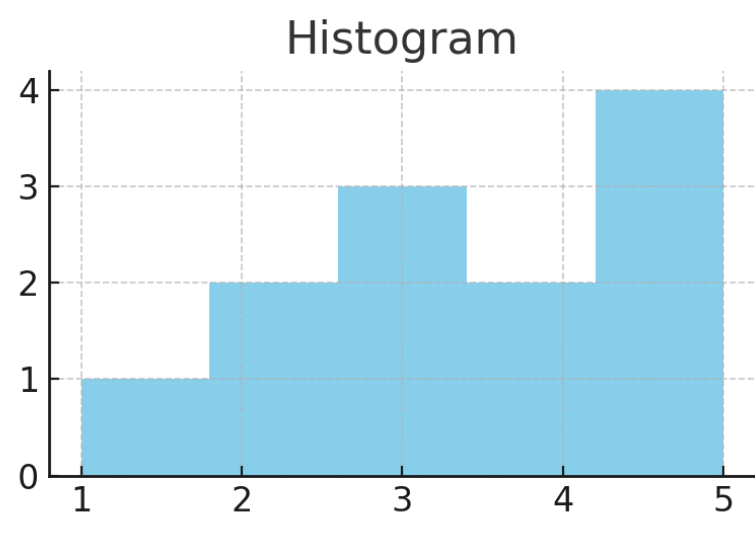
data = [1,2,2,3,3,3,4,4,5,5,5,5]

plt.hist(data, bins=5, color='skyblue')

plt.title("Histogram Example")

plt.show()

**Output:** Bars showing how many times each value or range occurs.



**SCATTER PLOT**

**Explanation:** Shows the relationship or correlation between two variables.  
**Input:** Lists or arrays for X and Y values.

**Code Example:**

x = [5, 7, 8, 7, 2, 17]

y = [99, 86, 87, 88, 100, 86]

plt.scatter(x, y, color='red')

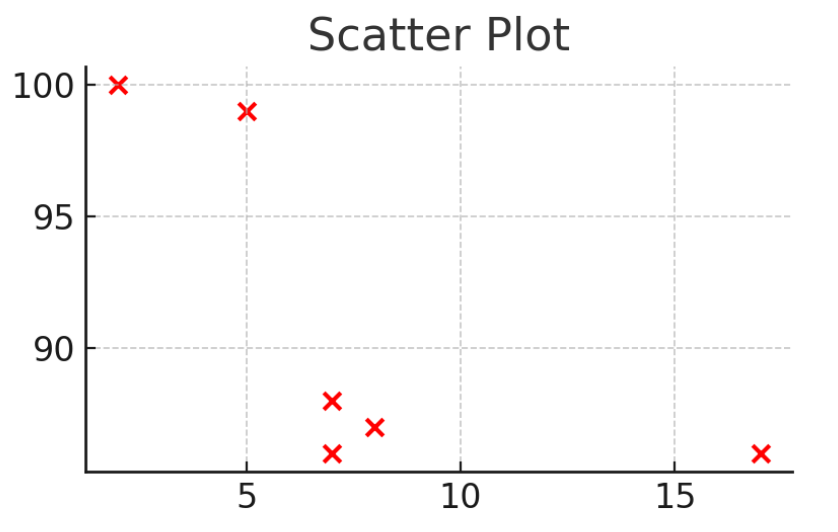
plt.title("Scatter Plot Example")

plt.xlabel("X-Axis")

plt.ylabel("Y-Axis")

plt.show()

**Output:** Individual points plotted on X-Y axes showing the data distribution.



**PIE CHART**

**Explanation:** Shows proportions or percentages of a whole.  
**Input:** List of values and labels.

**Code Example:**

sizes = [30, 25, 20, 25]

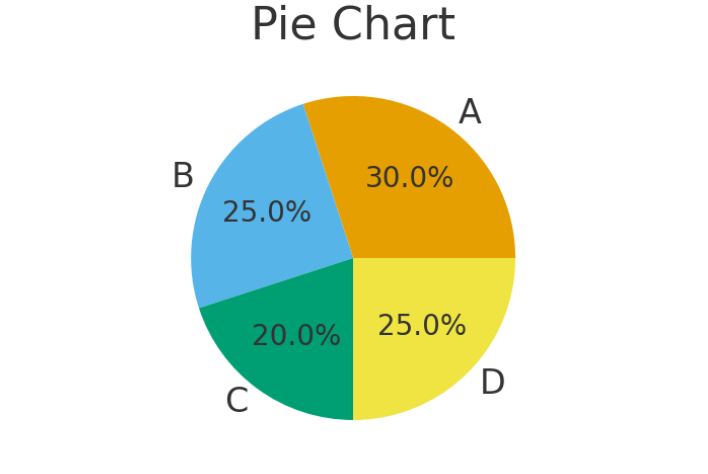
labels = ['A', 'B', 'C', 'D']

plt.pie(sizes, labels=labels, autopct='%1.1f%%')

plt.title("Pie Chart Example")

plt.show()

**Output:** Circular chart divided into slices proportional to the values.



**AREA CHART**

**Explanation:** Similar to a line plot but highlights the area under the curve.  
**Input:** Lists or arrays for X and Y values.

**Code Example:**

x = [1, 2, 3, 4, 5]

y = [2, 4, 6, 8, 10]

plt.fill\_between(x, y, color="skyblue", alpha=0.5)

plt.plot(x, y, color="blue")

plt.title("Area Chart Example")

plt.show()

**Output:** Line plot with the area under the line filled with color.

